**Effectiveness & Approach**

The design and intent of Program 5 was to take Course Media Playlist manager made in Program 4 and add the ability to have multiple different courses for the user to utilize. The program would allow the user to create their own playlists for a specific course as well as manage their courses as well. The main implementation of Program 5 was the facilitation of the multiple course concept via a balanced AVL tree.

This goal was achieved utilizing single inheritance with an Abstract Base Class named Media and two Derived Classes, Long video, and Short video and the use of dynamic binding. The Base Class would house the majority of the media’s information while the derived would focus on what made that type of media unique among its class. I believe this hierarchy made the execution of this program was better as it removed the need for the Derived classes to focus on anything but their specific role.

The program was broken down into four menus. The first menu was where the user could create and manage their courses. The second menu was where the user could create, remove, display all the types of media and operated as the main menu for the program and operated primarily on the fields that were in the base class, including adding comments to the media. The other two menus were utilized for use with the long and short video materials specifically and enabled the user to further customize their media by adding a URL, work cited page, and a quiz. The quiz feature was implement utilizing a “has a” relationship. This enabled either of the media types to be able to have a quiz that the user can create or not is they choose to.

**Object Oriented Programming**

The object oriented portion of the program was largely based in the implementation of program 4. The addition of balanced tree necessitated that the node class for the tree house its own media playlists. This was accomplished using a “has a” relationship where each node had a list object, and each list object held its own playlists.

**Data Structures**

The data structure was the main focus for this assignment. The data structure I chose to use to implement the program was an AVL Tree. The AVL Tree was implemented using a node class that housed the data for each course as well as making use of the list class made in program 4 to manage the creation of the tree.

The main focus of the implementation of the AVL Tree was ensuring that it was balanced after every insertion. This proved to be the largest challenged of the program and took a considerable amount of time and effort to ensure the tree was shifting correctly and data was not being misplaced. After an insertion would take place, the height of the node would be assessed and set and the balance factor for the tree would be determined. If the balance factor were not -1,0, or 1 the tree would initiate its shifting methods to rebalance itself.

The AVL Tree preformed exceptionally well as a method in storing the different courses. It allowed user to choose an option from the main menu and the program could respond efficiently and quickly to find the correct node and access its data. The AVL Tree had a complete implementation of Display All, Display One, and Remove All.

**Conclusion**

The success of Program 5 depended largely on the implementation of the AVL Tree and its ability to act as the data structure to hold the courses. I believe my implementation of the tree produced a correctly balanced tree as well as the proper structure for the user to be able to create different types of courses for the program.